Red Hat AMQ 2021.Q3

AMQ Streams 1.8 on RHEL リリースノート

AMQ Streams on Red Hat Enterprise Linux の使用

Last Updated: 2023-02-23

Red Hat AMQ 2021.Q3 AMQ Streams 1.8 on RHEL リリースノート

AMQ Streams on Red Hat Enterprise Linux の使用

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

本リリースノートには、AMQ Streams 1.8 リリースに含まれる新機能、改良された機能、修正、および問題に関する最新情報が含まれています。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージをご覧ください。

第1章 特長

本リリースで追加され、これまでの AMQ Streams リリースにはなかった機能は次のとおりです。

注記

本リリースで解決された改良機能とバグをすべて確認するには、AMQ Streams の Jira プロジェクト を参照してください。

1.1. KAFKA 2.8.0 のサポート

AMQ Streams は Apache Kafka バージョン 2.8.0 に対応するようになりました。

AMQ Streams は Kafka 2.8.0 を使用します。Red Hat によってビルドされた Kafka ディストリビューションのみがサポートされます。

アップグレードの手順は、「AMQ Streams and Kafka アップグレード」を参照してください。

詳細は、Kafka 2.7.0 および Kafka 2.8.0 のリリースノートを参照してください。

注記

Kafka 2.7.x は、AMQ Streams 1.8. にアップグレードする目的でのみサポートされます。

サポート対象バージョンの詳細は、Red Hat ナレッジベースの記事「Red Hat AMQ 7 Component Details Page」を参照してください。

Kafka 2.8.0 には ZooKeeper バージョン 3.5.9 が必要です。そのため、アップグレードに関するドキュメントで説明されているように、AMQ Streams 1.7 から AMQ Streams 1.8 にアップグレードする場合に ZooKeeper をアップグレードする必要があります。

警告

Kafka 2.8.0 では **自己管理モード** に早期にアクセスできます。このモードでは、Kafka が Raft プロトコルを使用して ZooKeeper なしに実行されます。**AMQ Streams では、自己管理モードがサポートされないことに注意してください。**

第2章 機能拡張

このリリースで改良された機能は次のとおりです。

2.1. KAFKA 2.8.0 で改良された機能

Kafka 2.8.0 に導入された改良機能の概要は『Kafka 2.8.0 Release Notes』を参照してください。

2.2. OAUTH 2.0 認証の改良

Audience および Scope の設定

oauth. **audience および oauth. scope プロパティーを設定し**、トークンの取得時にそれらの値をパラメーターとして渡すことができるようになりました。どちらのプロパティーも OAuth 2.0 認証リスナー設定で指定されます。

以下のシナリオで、このプロパティーを使用します。

- ブローカー間認証用のアクセストークンを取得する場合
- **clientId およびシークレット**を使用した OAuth 2.0 over PLAIN クライアント認証のクライアントの名前

これらのプロパティーは、クライアントがトークンとトークンのコンテンツを取得できるかどうかに影響します。リスナーによって課されるトークン検証ルールには影響を与えません。

oauth.audience および oauth.scope プロパティーの設定例

```
listener.name.client.oauthbearer.sasl.jaas.config=org.apache.kafka.common.security.oauthbearer.OAuthBearerLoginModule required \
  # ...
  oauth.token.endpoint.uri="https://AUTH-SERVER-ADDRESS/auth/realms/REALM-NAME/protocol/openid-connect/token" \
  oauth.scope=""SCOPE"" \
  oauth.audience="AUDIENCE" \
  oauth.check.audience="true" \
  # ...
```

承認サーバーは、JWT アクセストークンに **aud** (オーディエンス) クレームを提供することがあります。**oauth.check.audience="true"** を設定してオーディエンスチェックが有効な場合に、Kafka ブローカーは **aud** クレームにブローカーの **clientId** が含まれていないトークンを拒否します。オーディエンスチェックはデフォルトで無効になっています。

「Kafka ブローカーの OAuth 2.0 サポートの設定」を参照してください。

OAuth 2.0 over PLAIN でトークンエンドポイントを必要としない

oauth.token.endpoint.uri パラメーターは、OAuth 2.0 over PLAIN 認証に「クライアント ID および secret」メソッドを使用する場合に不要になりました。

トークンエンドポイント URI が指定された PLAIN リスナーが設定された OAuth 2.0 の例

```
listener.name.client.plain.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required \
  oauth.valid.issuer.uri="https://__AUTH-SERVER-ADDRESS__" \
```

```
  oauth.jwks.endpoint.uri="https://__AUTH-SERVER-ADDRESS__/jwks" \
  oauth.username.claim="preferred_username"  \
  oauth.token.endpoint.uri="http://__AUTH_SERVER__/auth/realms/__REALM__/protocol/openid-
connect/token" ;
```

oauth.token.endpoint.uri が指定されていない場合、リスナーは以下を処理します。

- **username** パラメーターをアカウント名として。
- **password** パラメーターを検証のために承認サーバーに渡される raw アクセストークンとして (OAUTHBEARER 認証の場合と同じ動作)。

OAuth 2.0 over PLAIN 認証の「long-lived access token」メソッドの動作は変更されません。このメソッドを使用する場合は **oauth.token.endpoint.uri** は必要ありません。

「OAuth 2.0 Kafka ブローカー設定」を参照してください。

第3章 テクノロジープレビュー

重要

テクノロジープレビューの機能は、Red Hat の実稼働環境のサービスレベルアグリーメント (SLA) ではサポートされず、機能的に完全ではないことがあるため、Red Hat はテクノロジープレビュー機能を実稼働環境に実装することは推奨しません。テクノロジープレビューの機能は、最新の技術をいち早く提供して、開発段階で機能のテストやフィードバックの収集を可能にするために提供されます。サポート範囲の詳細は、「テクノロジプレビュー機能のサポート範囲」を参照してください。

3.1. KAFKA STATIC QUOTA プラグインの設定

Kafka Static Quota プラグインを使用して、Kafka クラスターのブローカーにスループットおよびストレージの制限を設定します。バイトレートのしきい値およびストレージクォータを設定して、ブローカーと対話するクライアントに制限を設けることができます。

Kafka Static Quota プラグインの設定例

```
client.quota.callback.class= io.strimzi.kafka.quotas.StaticQuotaCallback
client.quota.callback.static.produce= 1000000
client.quota.callback.static.fetch= 1000000
client.quota.callback.static.storage.soft= 400000000000
client.quota.callback.static.storage.hard= 500000000000
client.quota.callback.static.storage.check-interval= 5
```

Setting limits on brokers using the Kafka Static Quota plugin を参照してください。

3.2. CRUISE CONTROL によるクラスターのリバランス

注記

Cruise Control は本リリースでもテクノロジープレビューですが、新たな改良が加えられました。

Cruise Control をデプロイして使用し、**最適化ゴール** (CPU、ディスク、ネットワーク負荷などに定義された制約) を使用し、Kafka をリバランスできます。バランス調整された Kafka クラスターでは、ワークロードがブローカー Pod 全体に均等に分散されます。

Cruise Control を使用すると、分散された Kafka クラスターを効率的に実行するための時間および労力を削減できます。

Cruise Control の zip ディストリビューションは、カスタマーポータル からダウンロードできます。Cruise Control をインストールするには、提供される Metrics Reporter を使用するように各 Kafka ブローカーを設定します。その後、最適化ゴールなどの Cruise Control プロパティーを設定し、指定のスクリプトを使用して Cruise Control を開始します。

Cruise Control サーバーは、Kafka クラスター全体に対して単一のマシンでホストされます。

Cruise Control の実行中に、REST API を使用して以下を行うことができます。

- 複数の最適化ゴールから、**ドライラン** の最適化プロポーザルを生成する。

- 最適化プロポーザルを開始して Kafka クラスターをリバランスする

異常検出、通知、独自ゴールの作成、トピックレプリケーション係数の変更などの、その他の Cruise Control の機能は現在サポートされていません。

「Cruise Control によるクラスターのリバランス」を参照してください。

3.2.1. テクノロジープレビューの改良

Cruise Control バージョン 2.5.59 では、最適化プロポーザルの計算を 10% 加速など、パフォーマンスが大幅に改善しました。

Red Hat カスタマーポータルから最新バージョンの zip 形式のディストリビューションをダウンロードできます。

カスタマーポータル を参照してください。

第4章 非推奨の機能

このリリースで非推奨となり、これまでの AMQ Streams リリースではサポートされていた機能は次のとおりです。

4.1. 非推奨となり削除された KAFKA 機能

本セクションでは、Apache Kafka プロジェクトで非推奨となり、削除される重要な機能を事前に報告します。

4.1.1. Kafka バージョン 3.0 で削除される予定の機能

Kafka バージョン 3.0 は、AMQ Streams の次回のメジャーリリースに同梱されます。

以下の表は、Kafka 2.x 以前で非推奨となり、Kafka 3.0 で**削除**される予定のメソッドやコンポーネントを示しています。このリストは包括的ではありません。

表4.1 Kafka 3.0 で削除される予定の非推奨の API メソッドおよびコンポーネント

API またはコンポーネント	課題へのリンク	説明
Admin API	KAFKA-12581	非推奨の Admin.electPreferredLeaders の削除
Admin API	KAFKA-6987	KafkaFuture を CompletableFuture で再実装 (KafkaFuture.Function は非推奨)
Admin client	KAFKA-12577	非推奨の **ConfigEntry** コンストラクターの削除
すべてのクライアント	KAFKA-12579	3.0 のクライアントからさまざまな非推奨メソッドを削除
すべてのクライアント	KAFKA-12600	クライアント設定 client. **dns.lookup**の非推奨の設定値の **デフォルト** を削除します。
すべてのクライアント	KAFKA-12578	非推奨のセキュリティークラス/メソッドの削除
Broker	KAFKA-12591	非推奨の **quota.producer.default** および **quota.consumer.default** の設定を削除します。
ブローカー	KAFKA-12592	非推奨の LogConfig.Compact の削除

API またはコンポーネント	課題へのリンク	説明
Broker	KAFKA-12590	非推奨の SimpleAclAuthorizer の削除
Broker	KAFKA-5905	PrincipalBuilder および DefaultPrincipalBuilder の削除
Common	KAFKA-12573	非推奨の **Metric#value** を削除
Consumer API	KAFKA-12637	非推奨の PartitionAssignor インターフェースの削除
Connect API	KAFKA-12482	非推奨の rest.host.name および rest.port Connect ワーカー設定の削除
Connect API	KAFKA-12945	3.0 で port、host.name、および関連設定を削除
Connect API	KAFKA-12717	内部コンバーター設定プロパティーの削除
Streams API	KAFKA-12574	eos-alpha の非推奨
Streams API	KAFKA-12808	StreamsMetrics で非推奨となったメソッドの削除
Streams API	KAFKA-7606	StreamsResetter から非推奨のオプションを削除
Streams API	KAFKA-12796	**streams-scala** の非推奨のクラスの削除
Streams API	KAFKA-12419	3.0 での Kafka Streams の非推奨 API の削除
Streams API	KAFKA-10434	WindowStore での非推奨メソッドの削除
Streams API	KAFKA-12449	非推奨の WindowStore#put の削除
Streams API	KAFKA-12813	ProcessorContext の非推奨のスケジュールメソッドの削除
Streams API	KAFKA-12809	Stores の非推奨のメソッドの削除

API またはコンポーネント	課題へのリンク	説明
Streams API	KAFKA-12814	非推奨メソッド StreamsConfig#getConsumerConfig の削除
Streams API	KAFKA-12313	default.windowed.serde.inner.class 設定を非推奨
Streams API	KAFKA-8372	非推奨の RocksDB#compactRange API の削除
Streams API	KAFKA-12584	非推奨の **Sum** および **Total** クラスの削除
Streams API	KAFKA-12683	非推奨の "UsePreviousTimeOnInvalidTimeStamp" の削除
Streams API	KAFKA-12810	非推奨の TopologyDescription.Source#topics の削除
Streams API	KAFKA-12630	非推奨の KafkaClientSupplier#getAdminClient の削除
Streams API	KAFKA-10046	非推奨の PartitionGrouper 設定は無視される
Streams API	KAFKA-12633	Remove deprecated "TopologyTestDriver#pipeInput / readOutput"
Streams API	KAFKA-12441	非推奨メソッド StreamsBuilder#addGlobalStore の削除
Streams API	KAFKA-12452	ProcessorContext#forward の非推奨オーバーロードの削除
Streams API	KAFKA-12450	ReadOnlyWindowStore から非推奨のメソッドを削除
Streams API	KAFKA-12880	3.0 で非推奨の Count と SampledTotal の削除

API またはコンポーネント	課題へのリンク	説明
Streams API	KAFKA-12451	WindowStore の長期ベースの読み取り操作の非推奨アノテーションを削除
Streams API	KAFKA-12568	非推奨の「KStream#groupBy/join」、「Joined#named」オーバーロードの削除
Streams API	KAFKA-12849	TaskMetadata を内部実装のインターフェースに移行
Streams API	KAFKA-7785	PartitionGrouper インターフェースと設定を削除し、DefaultPartitionGrouper を内部パッケージに移動。
Streams API	KAFKA-7106	Window 定義から segment/segmentInterval を削除
Streams API	KAFKA-8897	RocksDB のバージョンの増加
Streams API	KAFKA-12909	ユーザーに誤った left/outer stream-stream join の改善の選択を許可
Tools	KAFKA-8405	非推奨の **kafka-preferred-replica-election** コマンドの削除
Tools	KAFKA-12588	shell コマンドで非推奨の --zookeeper を削除

4.1.2. Kafka バージョン 4.0 で削除予定の Mirror Maker 1.0

Kafka バージョン 4.0 は、今後の AMQ Streams メジャーリリースに同梱される予定です。

以下の表は、Kafka 3.0 で非推奨となり、Kafka 4.0 で**削除**される予定の機能を示しています。

表4.2 Kafka 3.0 で非推奨となり Kafka 4.0 で削除される予定のコンポーネント

コンポーネント	課題へのリンク	概要
Mirror Maker 1.0	KAFKA-12436	MirrorMaker v1 の非推奨

第5章 修正された問題

AMQ Streams 1.8 on RHEL で修正された問題を、以下の表に示します。Kafka 2.8.0 で修正された問題の詳細は、『Kafka 2.8.0 Release Notes』を参照してください。

表5.1 修正された問題

課題番号	説明
ENTMQST-2453	理由がないため、**kafka-exporter** Pod を再起動します。
ENTMQST-2459	Kafka Exporter の実行により、CPU の使用率が高くなります。
ENTMQST-2511	ローリングアップデート中に Kafka Exporter の再起動を停止するようにヘルスチェックを微調整します。
ENTMQST-1529	サイズの大きいファイルの場合にファイルソースコネクターが停止します。

表5.2 CVE (Common Vulnerabilities and Exposures) の修正

課題番号	タイトル	説明
ENTMQST-3023	CVE-2021-34428 jetty-server: jetty: SessionListener により、セッションがログアウトの破損を非検証しなくなる。	jetty-server で SessionListener#sessionDestroyed() メソッドから例外が発生すると、セッション ID マネージャーでセッション ID が無効にならない不具合が見つかりました。クラスター化されたセッションと複数のコンテキストが含まれるデプロイメントでは、セッションが無効化されるのではなく、共有計算アプリケーションがログインされたままになっていました。この脆弱性では、データの整合性と機密性が最も懸念されます。
ENTMQST-2980	CVE-2021-28169 jetty-server: jetty: ConcatServlet および WelcomeFilter への要求が WEB-INF ディレクトリー内の保護されているリソースにアクセスできる。	-

課題番号	タイトル	説明
ENTMQST-2711	CVE-2021-21409 netty: content-length ヘッダーを使用した要求スマグリング。	Netty で不具合が発見されました。要求で endstream が true に設定された Http2HeaderFrame が 1 つ使用される場合には、content-length ヘッダーが正しく検証されない問題が発生します。要求がリモートピアにプロキシーされ、HTTP/1.1 に変換された場合に、リクエストスマグリングが発生します。この脆弱性では、整合性が最も懸念されます。
ENTMQST-2663	CVE-2021-27568 json-smart: 例外がキャッチされないことでクラッシュまたは情報公開が発生する	json-smart で不具合が発見されました。関数から例外がスローされ、キャッチされないと、ライブラリーを使用するプログラムがクラッシュしたり、機密情報を公開したりされる可能性があります。この脆弱性では、データの機密性およびシステムの可用性への影響が最も懸念されます。 OpenShift Container Platform (OCP) では、OCP メータリングスタックを構成する Hive/Presto/Hadoop コンポーネントには脆弱なバージョンの json-smart パッケージが同梱されます。OCP 4.6 リリース以降、メータリング製品は非推奨となったため [1]、影響を受けるコンポーネントは wontfix とマークされます。これは今後修正される可能性があります。 [1] https://docs.openshift.com/container-platform/4.6/release_notes/ocp-4-6-release-notes.html#ocp-4-6-metering-operator-deprecated

課題番号	タイトル	説明
ENTMQST-2647	CVE-2021-21295 netty: 検証がされないことが原因で HTTP/2 での要求スマグリングが可能に	バージョン 4.1.60.Final 以前の Netty (io.netty:netty-codec-http2) では、リクエストスマグリングを可能にする脆弱性があります。Content-Length ヘッダーが元の HTTP/2 要求にある場合に、伝播されるため、フィールドは **Http2MultiplexHandler** で検証されません。要求が HTTP/1.1 としてプロキシー化されていない場合は、問題ありません。要求が HTTP/2 ストリームとして受信され、**Http2StreamFrameToHttpObjectCodec** 経由で HTTP/1.1 ドメインオブジェクト (**HttpRequest**、**HttpContent** など) に変換され、HTTP/1.1 として子チャネルのパイプラインに送信され、HTTP/1.1 としてリモートピア経由でプロキシーされると、リクエストスマグリングが発生する可能性があります。
ENTMQST-2617	CVE-2021-21290 netty: ローカルシステムの一時ディレクトリーを介して情報が公開される。	Netty では、安全でない一時ファイルを使用する Unix のようなシステムに脆弱性があります。netty のマルチパートデコーダーが使用されており、ディスクへの一時的なアップロードが有効な場合に、ローカルシステムの一時ディレクトリーを使用してローカル情報を公開できます。unix のようなシステムでは、一時ディレクトリーは全ユーザー間で共有されます。そのため、ファイル/ディレクトリーのパーミッションを明示的に設定しない API を使用してこのディレクトリーに書き込むと、情報が開示される可能性があります。

課題番号	タイトル	説明
ENTMQST-2613	CVE-2020-13949 libthrift: 信頼できないペイロードを処理する時に DoS が発生する可能性がある。	libthrift で不具合が発見されました。Thrift を使用するアプリケーションは、ペイロードよりもサイズが大きいコンテナーを宣言するメッセージを受信しても、エラーが表示されません。これにより、悪意のある RPC クライアントで、短いメッセージを送信できるため、メモリーの割り当てが大きくなり、DoS につながる可能性があります。この脆弱性では、システムの可用性が最も懸念されます。
ENTMQST-1934	CVE-2020-9488 log4j: 証明書に SMTP アペンダーのホストに不一致がある場合に検証が正確に行われない。	-
ENTMQST-2910	CVE-2021-28163 jetty-server: jetty: Symlink ディレクトリーが webapp ディレクトリーの内容を公開する。	**${jetty.base}** または **${jetty.base}/webapps** ディレクトリーがシムリンクで、**${jetty.base}/webapps** ディレクトリーの内容が静的な Web アプリケーションとしてデプロイされている場合は、ダウンロードするディレクトリーのコンテンツを公開できます。この脆弱性では、データの機密性が最も懸念されます。
ENTMQST-2909	CVE-2021-28164 jetty-server: jetty: あいまいなパスで WEB-INF にアクセスできる。	Jetty では、デフォルトのコンプライアンスモードを使用すると、**%2e** または **%2e%2e** のセグメントが含まれる URI の要求で WEB-INF ディレクトリー内の保護されたリソースにアクセスできます。攻撃者はこの脆弱性を利用して、Web アプリケーションの実装に関する機密情報を得ることができます。

課題番号	タイトル	説明
ENTMQST-2908	CVE-2021-28165 jetty-server: jetty: サイズが大きく、無効な TLS フレームを受信すると、リソースが枯渇する、	HTTP/1.1、HTTP/2、または WebSocket のいずれかで SSL/TLS を使用する場合、サーバーは無効なサイズ (かつ 17408) TLS フレームを誤って処理し、CPU リソースの使用量が多くなる可能性があります。この脆弱性では、サービスの可用性が最も懸念されます。
ENTMQST-2867	CVE-2021-29425 commons-io: apache-commons-io: Apache Commons IO 2.2 から 2.6 へのパストラバーサルに制限がある。	-
ENTMQST-2821	CVE-2021-28168 jersey-common: jersey: システム一時ディレクトリーを介してローカル情報が公開される。	-

第6章 既知の問題

ここでは、AMQ Streams 1.8. の既知の問題について説明します。

6.1. LOG4J の SMTP アペンダー

AMQ Streams には、潜在的に脆弱なバージョンの log4j (**log4j-1.2.17.redhat-3**) が同梱されています。脆弱性は、デフォルト設定で AMQ Streams によって使用されない SMTP アペンダー機能にあります。

表6.1 CVE の問題

課題番号	説明
ENTMQST-1934	CVE-2020-9488 log4j: SMTP アペンダーのホスト不一致による証明書の不適切な検証 [amq-st-1]

回避策

SMTP アペンダーを使用している場合は、**mail.smtp.ssl.checkserveridentity** が **true** に設定されていることを確認します。

第7章 サポートされる統合製品

AMQ Streams 1.8 は、以下の Red Hat 製品との統合をサポートします。

Red Hat Single Sign-On 7.4 以降

OAuth 2.0 認証および OAuth 2.0 承認を提供します。

これらの製品によって AMQ Streams デプロイメントに導入可能な機能の詳細は、AMQ Streams 1.8 のドキュメントを参照してください。

その他のリソース

- Red Hat Single Sign-On でサポートされる構成

第8章 重要なリンク

- Red Hat AMQ 7 でサポートされる構成
- Red Hat AMQ 7 コンポーネントの詳細

Revised on 2021-12-18 13:43:08 +1000